DATA SCIENCE LABORATORY
LAB MANUAL

Semester	**:**	**VI**
Branch	**:**	**Artificial Intelligence**

J D College of Engineering, Nagpur

(Autonomous Institute)

S. No	Experiment
0	**TO STUDY THE R PROGRAMMING LABORATORY.**
1	**R AS CALCULATOR APPLICATION** a. Using with and without R objects on console b. Using mathematical functions on console c. Write an R script, to create R objects for calculator application and save in a specified location in disk.
2	**DESCRIPTIVE STATISTICS IN R** a. Write an R script to find basic descriptive statistics using summary, str, quartile function on mtcars& cars datasets. b. Write an R script to find subset of dataset by using subset (), aggregate () functions on iris dataset.
3	**READING AND WRITING DIFFERENT TYPES OF DATASETS** a. Reading different types of data sets (.txt, .csv) from Web and disk and writing in file in specific disk location. b. Reading Excel data sheet in R. c. Reading XML dataset in R.
4	**VISUALIZATIONS** **a.** Find the data distributions using box and scatter plot. **b.** Find the outliers using plot. **c.** Plot the histogram, bar chart and pie chart on sample data.
5	**CORRELATION AND COVARIANCE** a. Find the correlation matrix. b. Plot the correlation plot on dataset and visualize giving an overview of relationships among data on iris data. c. Analysis of covariance: variance (ANOVA), if data have categorical variables on iris data.
6	**REGRESSION MODEL** Import a data from web storage. Name the dataset and now do Logistic Regression to find out relation between variables that are affecting the admission of a student in a institute based on his or her GRE score, GPA obtained and rank of the student. Also check the model is fit or not. Require (foreign), require (MASS).
7	**MULTIPLE REGRESSION MODEL** Apply multiple regressions, if data have a continuous Independent variable. Apply on above dataset.
8	**REGRESSION MODEL FOR PREDICTION** Apply regression Model techniques to predict the data on above dataset.
9	**CLASSIFICATION MODEL** a. Install relevant package for classification. b. Choose classifier for classification problem. c. Evaluate the performance of classifier.
10	**CLUSTERING MODEL** a. Clustering algorithms for unsupervised classification.

	b. Plot the cluster data using R visualizations.

1. SYLLABUS:

DATA SCIENCE LABORATORY	
V Semester: CSE	
OBJECTIVES: **The course should enable the students to:** I. Understand the R Programming Language. II. Exposure on Solving of data science problems. III. Understand The classification and Regression Model.	
LIST OF EXPERIMENTS	
Week-1	**R AS CALCULATOR APPLICATION**
a. Using with and without R objects on console b. Using mathematical functions on console c. Write an R script, to create R objects for calculator application and save in a specified location in disk	
Week-2	**DESCRIPTIVE STATISTICS IN R**
a. Write an R script to find basic descriptive statistics using summary b. Write an R script to find subset of dataset by using subset ()	
Week-3	**READING AND WRITING DIFFERENT TYPES OF DATASETS**
a. Reading different types of data sets (.txt, .csv) from web and disk and writing in file in specific disk location. b. Reading Excel data sheet in R. c. Reading XML dataset in R.	
Week-4	**VISUALIZATIONS**
a. Find the data distributions using box and scatter plot. b. Find the outliers using plot. c. Plot the histogram, bar chart and pie chart on sample data	
Week-5	**CORRELATION AND COVARIANCE**
a. Find the correlation matrix. b. Plot the correlation plot on dataset and visualize giving an overview of relationships among data on iris data.	

c. Analysis of covariance: variance (ANOVA), if data have categorical variables on iris data	
Week-6	**REGRESSION MODEL**
Import a data from web storage. Name the dataset and now do Logistic Regression to find out relation between variables that are affecting the admission of a student in a institute based on his or her GRE score, GPA obtained and rank of the student. Also check the model is fit or not. require (foreign), require(MASS).	
Week-7	**MULTIPLE REGRESSION MODEL**
Apply multiple regressions, if data have a continuous independent variable. Apply on above dataset.	
Week-8	**REGRESSION MODEL FOR PREDICTION**
Apply regression Model techniques to predict the data on above dataset	
Week-9	**CLASSIFICATION MODEL**
a. Install relevant package for classification. b. Choose classifier for classification problem. c. Evaluate the performance of classifier.	
Week-10	**CLUSTERING MODEL**
a. Clustering algorithms for unsupervised classification. b. Plot the cluster data using R visualizations.	
Reference Books:	
Yanchang Zhao, “R and Data Mining: Examples and Case Studies”, Elsevier, 1st Edition, 2012	
Web References:	
1.http://www.r-bloggers.com/how-to-perform-a-logistic-regression-in-r/ 2.http://www.ats.ucla.edu/stat/r/dae/rreg.htm 3.http://www.coastal.edu/kingw/statistics/R-tutorials/logistic.html 4. http://www.ats.ucla.edu/stat/r/data/binary.csv	
SOFTWARE AND HARDWARE REQUIREMENTS FOR 18 STUDENTS: SOFTWARE: R Software , R Studio Software HARDWARE: 18 numbers of Intel Desktop Computers with 4 GB RAM	

INDEX

Practical No-0

Aim: - Introduction to R- Programming Lab.

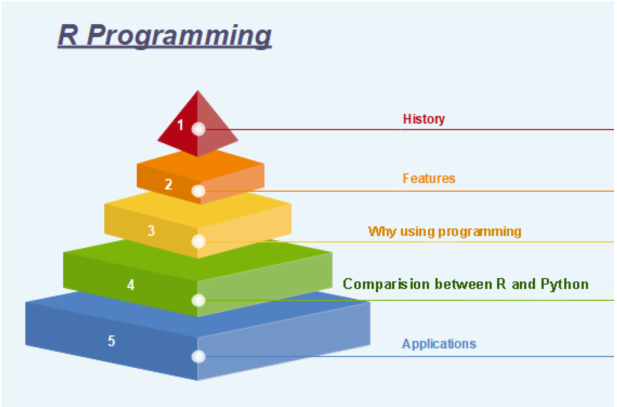
Theory: -

What is R Programming

"R is an interpreted computer programming language which was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand." The R Development Core Team currently develops R. It is also a software environment used to analyze statistical information, graphical representation, reporting, and data modeling. R is the implementation of the S programming language, which is combined with lexical scoping semantics.

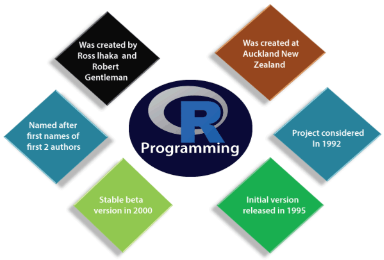
R not only allows us to do branching and looping but also allows to do modular programming using functions. R allows integration with the procedures written in the C, C++, .Net, Python, and FORTRAN languages to improve efficiency.

In the present era, R is one of the most important tool which is used by researchers, data analyst, statisticians, and marketers for retrieving, cleaning, analyzing, visualizing, and presenting data.



History of R Programming

The history of R goes back about 20-30 years ago. R was developed by Ross lhaka and Robert Gentleman in the University of Auckland, New Zealand, and the R Development Core Team currently develops it. This programming language name is taken from the name of both the developers. The first project was considered in 1992. The initial version was released in 1995, and in 2000, a stable beta version was released.



Features of R programming

R is a domain-specific programming language which aims to do data analysis. It has some unique features which make it very powerful. The most important arguably being the notation of vectors. These vectors allow us to perform a complex operation on a set of values in a single command. There are the following features of R programming:

1. It is a simple and effective programming language which has been well developed.
2. It is data analysis software.
3. It is a well-designed, easy, and effective language which has the concepts of user-defined, looping, conditional, and various I/O facilities.
4. It has a consistent and incorporated set of tools which are used for data analysis.
5. For different types of calculation on arrays, lists and vectors, R contains a suite of operators.

6. It provides effective data handling and storage facility.
7. It is an open-source, powerful, and highly extensible software.
8. It provides highly extensible graphical techniques.
9. It allows us to perform multiple calculations using vectors.
10. R is an interpreted language.

Why use R Programming?

There are several tools available in the market to perform data analysis. Learning new languages is time taken. The data scientist can use two excellent tools, i.e., R and Python. We may not have time to learn them both at the time when we get started to learn data science. Learning statistical modeling and algorithm is more important than to learn a programming language. A programming language is used to compute and communicate our discovery.

The important task in data science is the way we deal with the data: clean, feature engineering, feature selection, and import. It should be our primary focus. Data scientist job is to understand the data, manipulate it, and expose the best approach. For machine learning, the best algorithms can be implemented with R. **Keras** and **TensorFlow** allow us to create high-end machine learning techniques. R has a package to perform **Xgboost**. Xgboost is one of the best algorithms for **Kaggle competition**.

R communicate with the other languages and possibly calls Python, Java, C++. The big data world is also accessible to R. We can connect R with different databases like **Spark** or **Hadoop**.

In brief, R is a great tool to investigate and explore the data. The elaborate analysis such as clustering, correlation, and data reduction are done with R.

Comparison between R and Python

Data science deals with identifying, extracting, and representing meaningful information from the data source. R, Python, SAS, SQL, Tableau, MATLAB, etc. are the most useful tools for data science. R and Python are the most used ones. But still, it becomes confusing to choose the better or the most suitable one among the two, R and Python.

Comparison Index	R	Python
Overview	"R is an interpreted computer programming language which was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand ." The R Development Core Team currently develops R. R is also a software environment which is used to analyze statistical information, graphical representation, reporting, and data modeling.	Python is an Interpreted high-level programming language used for general-purpose programming. Guido Van Rossum created it, and it was first released in 1991. Python has a very simple and clean code syntax. It emphasizes the code readability and debugging is also simple and easier in Python.
Specialties for data science	R packages have advanced techniques which are very useful for statistical work. The CRAN text view is provided by many useful R packages. These packages cover everything from Psychometrics to Genetics to Finance.	For finding outliers in a data set both R and Python are equally good. But for developing a web service to allow peoples to upload datasets and find outliers, Python is better.
Functionalities	For data analysis, R has inbuilt functionalities	Most of the data analysis functionalities are not inbuilt. They are available through packages like Numpy and Pandas
Key domains of	Data visualization is a key aspect of analysis.	Python is better for deep learning because

application	R packages such as ggplot2, ggvis, lattice, etc. make data visualization easier.	Python packages such as Caffe, Keras, OpenNN, etc. allows the development of the deep neural network in a very simple way.
Availability of packages	There are hundreds of packages and ways to accomplish needful data science tasks.	Python has few main packages such as viz, Sccikit learn, and Pandas for data analysis of machine learning, respectively.

Applications of R

There are several-applications available in real-time. Some of the popular applications are as follows:

- Facebook
- Google
- Twitter
- HRDAG
- Sunlight Foundation
- Real Climate
- NDAA
- XBOX ONE
- ANZ
- FDA

Prerequisite

R programming is used for statistical information and data representation. So it is required that we should have the knowledge of statistical theory in mathematics. Understanding of different types of graphs for data representation and most important is that we should have prior knowledge of any programming.

Conclusion: - Thus we have studied R-programming.

Experiment 1 - <u>R AS CALCULATOR APPLICATION</u>

a. Using without R objects on console

```
> 2587+2149

Outpu

t:-

[1]

4736

> 287954-135479
Output:
- [1]
152475
>
257*52
[1]
13364

> 257/21

Output:-

[1]

12.2381
```

Using with R objects on console:

```
>A=1000
>B=2000
>c=A+B
```

```
>c

Outpu

t:-

[1]

3000
```

b. Using mathematical functions on console

```
>a=100

>class(a)

[1] "numeric"

>b=500

>c=a-b

>class(b)

[1] "numeric"

>sum<a-b
```

```
[1] FALSE

>sum

[1] -400
```

c. **Write an R script, to create R objects for calculator application andsave in a specified location in disk.**

```
getwd()

[1] "C:/Users/Administrator/Documents"
```

```
getwd()
my_data<-data.frame( Name=c("Alice","Bob", "Charlie"),
                     Age= c( 25,30,22),
                     Score= c(95,89,75)
)
write.csv(my_data,"output_file.csv", row.names= FALSE)
```

```
getwd()
my_data<-data.frame( Name=c("Alice","Bob", "Charlie", "sUJATA"),
                     Age= c( 25,30,22, 30),
                     Score= c(95,89,75,99)
)
my_data<-read.csv("output_file.csv")
head(my_data)
```

Experiment 2 - DESCRIPTIVE STATISTICS IN R

a. Write an R script to find basic descriptive statistics using summary, str, quartile function on mtcars& cars datasets.

```
>mtcars
mpgcyldisphp drat    wtqsec vs am gear carb
Mazda RX4            21.0    6 160.0 110 3.90 2.620 16.46 0 1         4
4
Mazda RX4 Wag        21.0    6 160.0 110 3.90 2.875 17.02 0 1         4
4
Datsun 710           22.8    4 108.0 93 3.85 2.320 18.61 1 1          4
1
Hornet 4 Drive       21.4    6 258.0 110 3.08 3.215 19.44 1 0         3
1
Hornet Sportabout    18.7    8 360.0 175 3.15 3.440 17.02 0 0         3
2
Valiant              18.1    6 225.0 105 2.76 3.460 20.22 1 0         3
1
Duster 360           14.3    8 360.0 245 3.21 3.570 15.84 0 0         3
4
Merc 240D            24.4    4 146.7 62 3.69 3.190 20.00 1 0          4
2
Merc 230             22.8    4 140.8 95 3.92 3.150 22.90 1 0          4
2
Merc 280             19.2    6 167.6 123 3.92 3.440 18.30 1 0         4
4
Merc 280C            17.8    6 167.6 123 3.92 3.440 18.90 1 0         4
4
Merc 450SE           16.4    8 275.8 180 3.07 4.070 17.40 0 0         3
3
Merc 450SL           17.3    8 275.8 180 3.07 3.730 17.60 0 0         3
3
Merc 450SLC          15.2    8 275.8 180 3.07 3.780 18.00 0 0         3
3
Cadillac Fleetwood 10.4     8 472.0 205 2.93 5.250 17.98 0 0          3
4
Lincoln Continental 10.4    8 460.0 215 3.00 5.424 17.82    0   0     3
4
Chrysler Imperial    14.7    8 440.0 230 3.23 5.345 17.42 0 0         3
4
Fiat 128             32.4    4 78.7 66 4.08 2.200 19.47 1 1          4
1
Honda Civic          30.4    4 75.7 52 4.93 1.615 18.52 1 1          4
2
Toyota Corolla       33.9    4 71.1 65 4.22 1.835 19.90 1 1          4
1
```

```
Toyota Corona         21.5    4 120.1 97 3.70 2.465 20.01 1 0          3
1
Dodge Challenger      15.5    8 318.0 150 2.76 3.520 16.87 0 0         3
2
AMC Javelin           15.2    8 304.0 150 3.15 3.435 17.30 0 0         3
```

2

```
13.3    8 350.0 245 3.73 3.840 15.41    0   0     3
19.2    8 400.0 175 3.08 3.845 17.05    0   0     3

27.3    4   79.0   66  4.08  1.935  18.90   1   1     4
26.0    4  120.3   91  4.43  2.140  16.70   0   1     5
30.4    4   95.1  113  3.77  1.513  16.90   1   1     5
15.8    8  351.0  264  4.22  3.170  14.50   0   1     5
19.7    6  145.0  175  3.62  2.770  15.50   0   1     5
15.0    8  301.0  335  3.54  3.570  14.60   0   1     5
21.4    4  121.0  109  4.11  2.780  18.60   1   1     4
```

Camaro
Z28 4
Pontiac
Firebird 2
Fiat
X1-9 1
Porsche 914-2
2
Lotus
Europa 2
Ford
Pantera L 4
Ferrari
Dino 6
Maserati
Bora 8
Volvo
142E 2

```
>summary(mtcars)

mpgcyldisphp              drat
Min.:10.40   Min.   :4.000   Min.   : 71.1   Min.: 52.0   Min.:2.760
 1st Qu.:15.43    1st Qu.:4.000    1st Qu.:120.8    1st Qu.: 96.5
 1st
Qu.:3.080
Median :19.20   Median :6.000   Median :196.3   Median :123.0   Median
:3.695
 Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7   Mean
:3.597
 3rd Qu.:22.80    3rd Qu.:8.000    3rd Qu.:326.0    3rd Qu.:180.0
3rd Qu.:3.920
 Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0   Max.
:4.930
wtqsec             vs                am               gear
```

```
Min.:1.513    Min.    :14.50    Min.    :0.0000    Min.    :0.0000    Min.
:3.000
 1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000   1st Qu.:0.0000
1st Qu.:3.000
Median :3.325     Median :17.71     Median :0.0000     Median :0.0000
Median :4.000
 Mean    :3.217    Mean    :17.85    Mean    :0.4375    Mean    :0.4062
Mean   :3.688
 3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000   3rd Qu.:1.0000
3rd Qu.:4.000
 Max.    :5.424    Max.    :22.90    Max.    :1.0000    Max.    :1.0000
Max.   :5.000
carb
Min.:1.
000
 1st Qu.:2.000
Median :2.000
 Mean
 :2.812  3rd
 Qu.:4.000
 Max.
 :8.000


>str(mtcars)
```

```
'data.frame': 32 obs. of  11 variables:
$ mpg :num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl :num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $  hp :num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $  wt :num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $  vs :num  0 0 1 1 0 1 0 1 1 1 ...
 $  am :num  1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num  4 4 1 1 2 1 4 2 2 4 ...

>quantile(mtcars$mpg)

  0%     25%     50%     75%    100%
     10.400 15.425 19.200 22.800
                          33.900
```

```
1       4     2
2       4    10
3       7     4
4       7    22
5       8    16
6       9    10
7      10    18
8      10    26
9      10    34
10     11    17
11     11    28
12     12    14
13     12    20
14     12    24
15     12    28
16     13    26
17     13    34
18     13    34
19     13    46
20     14    26
21     14    36
22     14    60
23     14    80
24     15    20
25     15    26
26     15    54
27     16    32
28     16    40
29     17    32
30     17    40
31     17    50
32     18    42
33     18    56
34     18    76
35     18    84
36     19    36
  >cars
```

speedd

ist

```
37    19   46
38    19   68
39    20   32
40    20   48
41    20   52
42    20   56
43    20   64
44    22   66
45    23   54
46    24   70
47    24   92
48    24   93
49    24  120
50    25   85
```

```
>summary(cars)

speeddist
Min.: 4.0   Min.   : 2.00
 1st Qu.:12.0   1st Qu.: 26.00
Median :15.0   Median : 36.00
 Mean  :15.4  Mean    :  42.98
 3rd   Qu.:19.0   3rd   Qu.:
 56.00   Max.   :25.0   Max.
 :120.00

>class(cars)
[1] "data.frame"

>dim(ca

rs) [1]

50 2

>str(cars)

'data.frame': 50 obs. of  2 variables:
 $ speed: num 4 4 7 7 8 9 10 10 10 11 ...
 $ dist :num 2 10 4 22 16 10 18 26 34 17 ...
```

>quantile(cars$speed)

```
  0%   25%   50%   75%  100%
   4    12    15    19    25
```

b. Write an R script to find subset of dataset by using subset (), aggregate () functions on iris dataset.

>aggregate(. ~ Species, data = iris, mean)

Output:

	Species	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	setosa	5.006	3.428	1.462	0.246
2	versicolor	5.936	2.770	4.260	1.326
3	virginica	6.588	2.974	5.552	2.026

>subset(iris,iris$Sepal.Length==5.0)

Output:

```
Sepal.LengthSepal.WidthPetal.LengthPetal.WidthSpecies
5              5         3.6          1.4          0.2     setosa
8              5         3.4          1.5          0.2     setosa
26             5         3.0          1.6          0.2     setosa
27             5         3.4          1.6          0.4     setosa
36             5         3.2          1.2          0.2     setosa
41             5         3.5          1.3          0.3     setosa
44             5         3.5          1.6          0.6     setosa
50             5         3.3          1.4          0.2     setosa
61             5         2.0          3.5          1.0 versicolor
94             5         2.3          3.3          1.0 versicolor
```

Experiments 3 - READING AND WRITING DIFFERENT TYPES OF DATASETS

a. Reading different types of data sets (.txt, .csv) from web and disk and writing in file in specific disk location.

```
library(utils)
data<-
read.csv("input.csv")
data
```

Output :-

```
id, name,      salary, start_date,       dept
1     1   Rick      623.30    2012-01-01      IT
2     2   Dan       515.20    2013-09-23      Operations
3     3   Michelle 611.00    2014-11-15      IT
4     4   Ryan      729.00    2014-05-11      HR
5    NA   Gary      843.25    2015-03-27      Finance
6     6   Nina      578.00    2013-05-21      IT
7     7   Simon     632.80    2013-07-30      Operations
8     8   Guru      722.50    2014-06-17      Finance
```

```
data<- read.csv("input.csv")

print(is.data.frame(data)
) print(ncol(data))
print(nrow(data))
```

Output:-

```
[1] TRUE
[1] 5
[1] 8
```

```
# Create a data frame.
data<-
read.csv("input.csv")

# Get the max salary from data
frame. sal<- max(data$salary)
sal
```

Outp

ut:-

[1]

843.2

5

Create a data frame.
data<-
read.csv("input.csv")

Get the max salary from data
frame. sal<- max(data$salary)

Get the person detail having max
salary. retval<- subset(data, salary ==
max(salary)) retval

Output:-

id name salary start_datedept
5 NA Gary 843.25 2015-03-27

Finance Get all the people working in IT

department

Create a data frame.
data<-
read.csv("input.csv")

retval<- subset(data, dept ==
"IT") retval

Output:-

id name salary start_datedept
1 1 Rick 623.3 2012-01-01 IT
3 3 Michelle 611.0 2014-11-15 IT
6 6 Nina 578.0 2013-05-21 IT

#Create a data frame.

data<- read.csv("input.csv")
retval<- subset(data, as.Date(start_date) >as.Date("2014-01-01"))

Write filtered data into a new file.

write.csv(retval,"output.csv")
newdata<-
read.csv("output.csv")
newdata

Output:-

X id name salary start_datedept
1 3 3 Michelle 611.00 2014-11-15 IT

```
2 4      4    Ryan       729.00 2014-05-11       HR
3 5     NA    Gary       843.25 2015-03-27      Finance
4 8      8    Guru       722.50 2014-06-17      Finance
```

b. Reading Excel data sheet in R.

```
install.packages("xl
sx") library("xlsx")
data<- read.xlsx("input.xlsx", sheetIndex
= 1) data
```

Output:-

```
id, name,      salary, start_date,      dept
1     1    Rick      623.30    2012-01-01      IT
2     2    Dan       515.20    2013-09-23      Operations
3     3    Michelle 611.00    2014-11-15      IT
4     4    Ryan      729.00    2014-05-11      HR
5    NA    Gary      843.25    2015-03-27      Finance
6     6    Nina      578.00    2013-05-21      IT
7     7    Simon     632.80    2013-07-30      Operations
8     8    Guru      722.50    2014-06-17      Finance
```

c. Reading XML dataset in R.

```
install.packages("X
ML") library("XML")
library("methods")
result<- xmlParse(file =
"input.xml") result
```

Output:-

```
1
   Ri
   c
   k
   6
   2
   3.
   3
   1/1/2012
   IT


   2
   D
   a
   n
   5
   1
```

5.
2
9/23/2013

Operations

3
Mich
elle
611
11/15/2014
IT

4
R
y
a
n
7
2
9
5/11/2014
HR

5
Ga
ry
84
3.2
5
3/27/2015
Finance

6
N
i
n
a
5
7
8
5/21/2013
IT

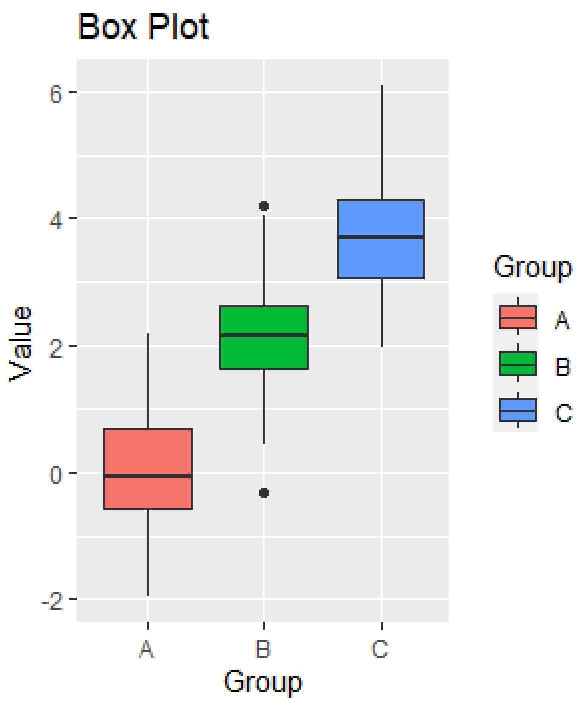
7
Si
m
on
63
2.
8
7/30/2013
Operations

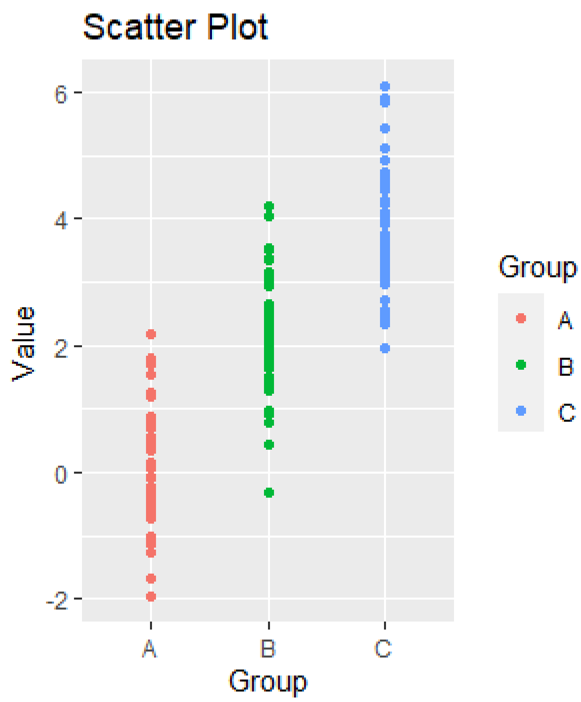
8
G
u
r
u
7
2
2.
5
6/17/2014
Finance

Experiment 4 – VISUALIZATIONS

a. Find the data distributions using box and scatter plot.

```
 # Install necessary packages if not already installed
# install.packages("ggplot2")

# Load the ggplot2 library
library(ggplot2)

# Generate random data for demonstration purposes
set.seed(123)
data <- data.frame(
  Group = rep(c("A", "B", "C"), each = 50),
  Value = c(rnorm(50), rnorm(50, mean = 2), rnorm(50, mean = 4))
)

# Create a box plot
box_plot <- ggplot(data, aes(x = Group, y = Value, fill = Group)) +
  geom_boxplot() +
  labs(title = "Box Plot", x = "Group", y = "Value")

# Create a scatter plot
scatter_plot <- ggplot(data, aes(x = Group, y = Value, color = Group)) +
  geom_point() +
  labs(title = "Scatter Plot", x = "Group", y = "Value")

# Display the plots side by side
library(gridExtra)
  grid.arrange(box_plot, scatter_plot, ncol = 2)
```





b. Plot the histogram, bar chart and pie chart on sample data.

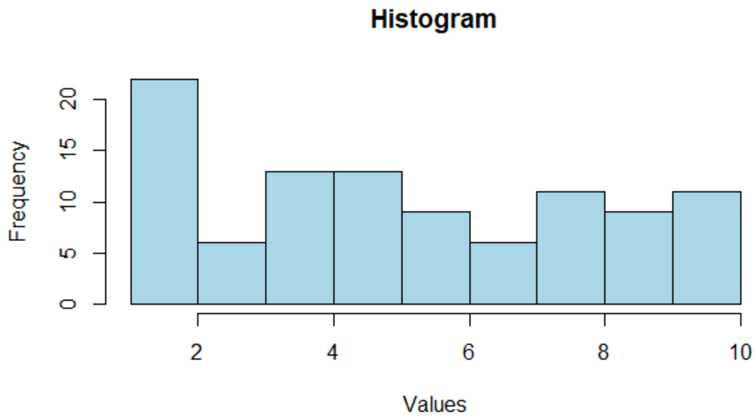
pie chart: -

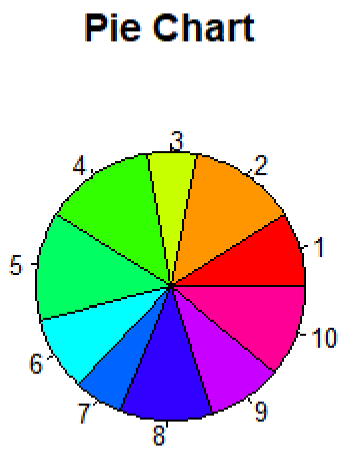
# Generating sample data
set.seed(42)
data <- sample(1:10, 100, replace = TRUE)

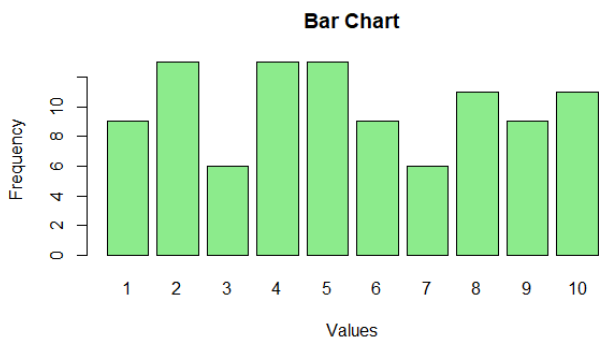
# Plotting histogram
hist(data, main = "Histogram", xlab = "Values", ylab = "Frequency", col = "lightblue", border = "black")

# Plotting bar chart
barplot(table(data), main = "Bar Chart", xlab = "Values", ylab = "Frequency", col = "lightgreen", border = "black")

# Plotting pie chart
pie(table(data), main = "Pie Chart", col = rainbow(length(unique(data))), cex = 0.8)


Histogram
Frequency
0
5
10
15
20
2
4
6
8
10
Values


Pie Chart
1
2
3
4
5
6
7
8
9
10


Bar Chart
Frequency
0
2
4
6
8
10
1
2
3
4
5
6
7
8
9
10
Values

Experiment 5:

PROBLEM DEFINATION:

a) How to find a corelation matrix and plot the correlation on iris data set

SOURCE CODE:

```
d<-data.frame(x1=rnorm(!0),x2=rnorm(10),x3=rnorm(10))
cor(d)
m<-cor(d) #get
correlations
library(„corrplot‟)
corrplot(m,method="squa
re")
x<-matrix(rnorm(2),,nrow=5,ncol
=4)
y<-matrix(rnorm(15),nrow=5,nco
l=3) COR<-cor(x,y)
COR
```

PROBLEM DEFINATION:

b) Plot the correlation plot on dataset and visualize giving an overview of relationships among data on iris data.

SOURCE CODE:

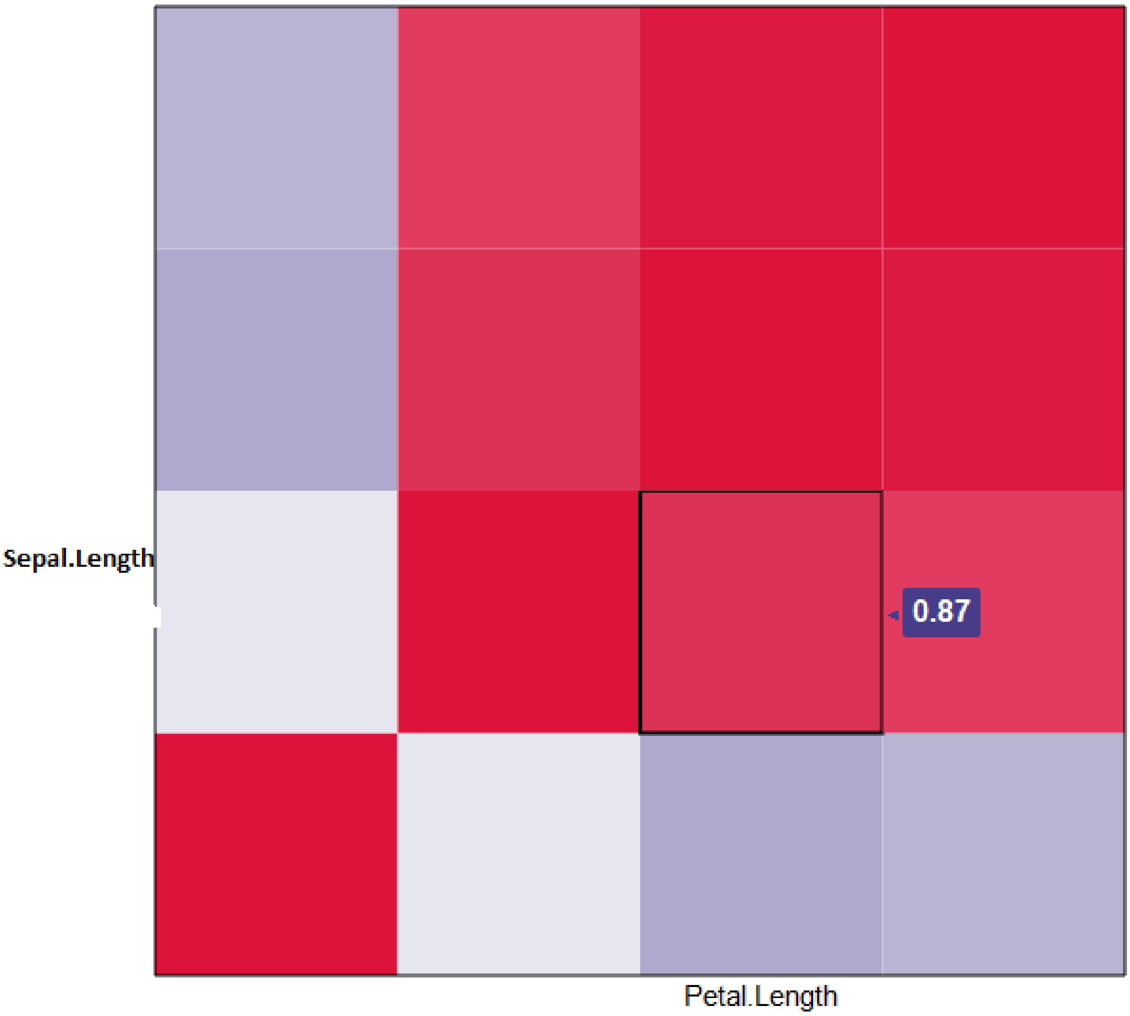
```
Image(x=seq(dim(x)[2])
Y<-seq(dim(y)[2])
Z=COR,xlab="xcolumn",ylab="y
column") Library(gtlcharts)
Data(iris)
Iris$species<-N
ULL
Iplotcorr(iris,reoder=TRUE
```

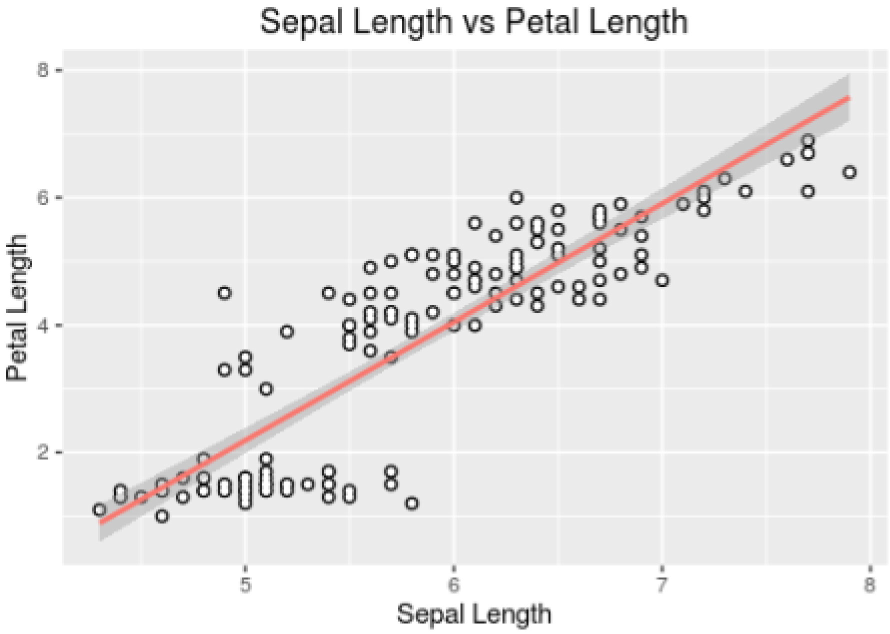
PROBLEM DEFINATION:

c) Analysis of covariance: variance (ANOVA), if data have categorical variables on iris data. SOURCE CODE:

```
library(ggpl
ot2)
data(iris)
str(iris)
ggplot(data=iris,aes(x=sepal.length,y=petal.length))+geom_point(size=2,colour="black")+ge
om_
point(size=1,colour="white")+geom_smooth(aes(colour="black"),method="lm‟)+ggtitle("se
pal.l
engthvspetal.length")+xlab("sepal.length")+ylab("petal.length")+these(legend.position="non
e")
```

OUTPUT:





Sepal Length vs Petal Length
Petal Length
Sepal Length
2
4
6
8
5
6
7
8

Experiment no. 6

PROBLEM DEFINATION:

REGRESSION MODEL: Linear regression is a statistical method used to model the relationship between a dependent variable (often denoted as Y) and one or more independent variables (often denoted as X). The relationship is modeled as a linear equation:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \varepsilon$$

Where:

- Y is the dependent variable (the variable we are trying to predict).
- $X_1, X_2, \ldots, X_n$ are the independent variables (also called predictors or features).
- β_0 is the intercept (the value of Y when all X are zero).
- $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients (the change in Y for a one-unit change in each X).
- ε is the error term (the difference between the predicted Y and the actual Y).

The goal of linear regression is to estimate the coefficients $\beta_0, \beta_1, \ldots, \beta_n$ that minimize the difference between the observed values of Y and the values predicted by the linear equation.

Linear regression can be used for various purposes, such as:

- Predictive modelling: Predicting the value of the dependent variable based on the values of the independent variables.
- Relationship analysis: Determining the strength and direction of the relationship between variables.
- Estimating the effect of predictors: Assessing the impact of changes in the independent variables on the dependent variable.

It's called "linear" regression because the relationship between the dependent and independent variables is assumed to be linear. However, it's worth noting that the method can be extended to model non-linear relationships by including transformations of the predictors (e.g., polynomial regression) or by using more complex techniques like generalized linear models.

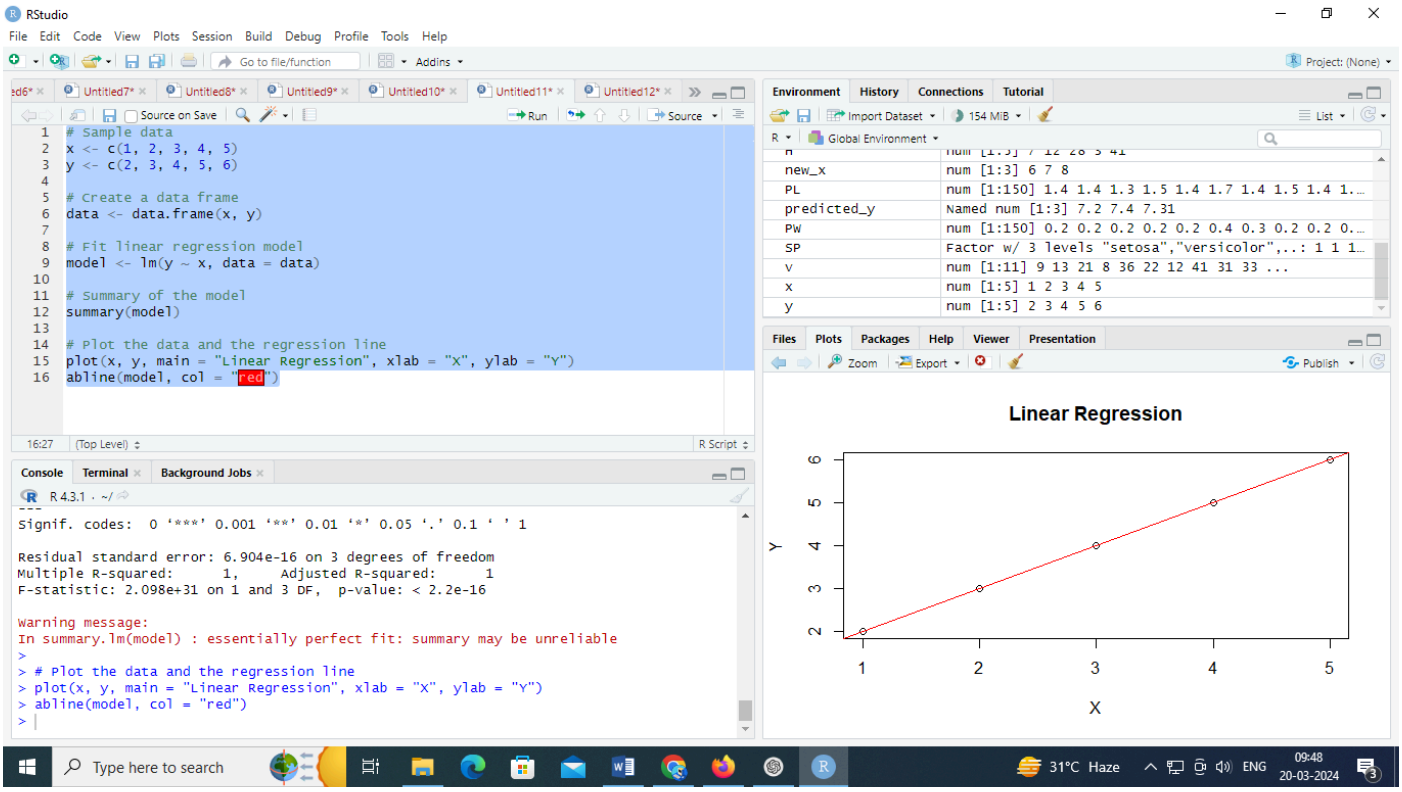
A) write a code in R Programming Linear REGRESSION MODEL.

SOURCE CODE:

```
# Sample data
x <- c(1, 2, 3, 4, 5)
y <- c(2, 3, 4, 5, 6)

# Create a data frame
data <- data.frame(x, y)

# Fit linear regression model
model <- lm(y ~ x, data = data)

# Summary of the model
summary(model)

# Plot the data and the regression line
```

```
plot(x, y, main = "Linear Regression", xlab = "X", ylab = "Y")
abline(model, col = "red")
```

This code creates a simple linear regression model where y is regressed on x. You can replace the x and y vectors with your own data. The lm() function is used to fit the linear regression model, and summary() provides a summary of the model's statistics. Finally, the plot() function is used to visualize the data along with the fitted regression line.



B) write a code in R Programming Polynomial REGRESSION MODEL.

REGRESSION MODEL: Polynomial regression is a type of regression analysis in which the relationship between the independent variable X and the dependent variable Y is modeled as an n-th degree polynomial. Unlike linear regression, which fits a straight line to the data, polynomial regression fits a curve.

The general form of a polynomial regression model of degree n is:

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \ldots + \beta_n X^n + \varepsilon$$

Where:

- Y is the dependent variable.
- X is the independent variable.

$\beta_0, \beta_1, \ldots, \beta_n$ are the coefficients.

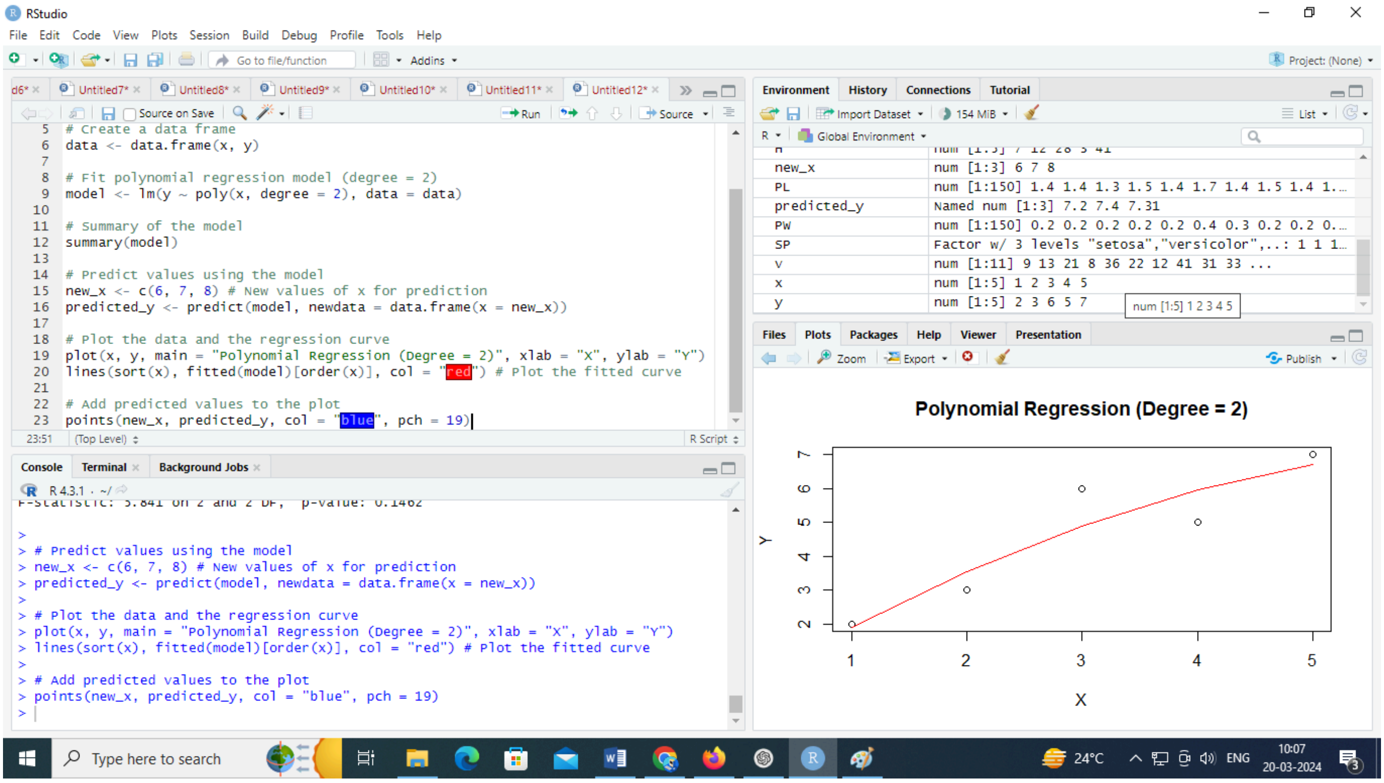
- ε is the error term.

In polynomial regression, the goal is to estimate the coefficients $\beta_0, \beta_1, \ldots, \beta_n$ that minimize the difference between the observed values of Y and the values predicted by the polynomial equation.

```
# Sample data
x <- c(1, 2, 3, 4, 5)
y <- c(2, 3, 6, 5, 7)

# Create a data frame
data <- data.frame(x, y)

# Fit polynomial regression model (degree = 2)
model <- lm(y ~ poly(x, degree = 2), data = data)

# Summary of the model
summary(model)

# Predict values using the model
new_x <- c(6, 7, 8) # New values of x for prediction
predicted_y <- predict(model, newdata = data.frame(x = new_x))

# Plot the data and the regression curve
plot(x, y, main = "Polynomial Regression (Degree = 2)", xlab = "X", ylab = "Y")
lines(sort(x), fitted(model)[order(x)], col = "red") # Plot the fitted curve

# Add predicted values to the plot
points(new_x, predicted_y, col = "blue", pch = 19)
```

In this code:

We create some sample data x and y.
We fit a polynomial regression model of degree 2 using the poly() function.
predict () is used to predict new values of y for given x values.
We plot the original data points and the fitted polynomial curve.
You can adjust the degree of the polynomial by changing the degree parameter in the poly() function.



Experiment No.7: CLASSIFICATION MODEL

PROBLEM DEFINATION:

Apply multiple regressions, if data have a continuous independent variable. Apply on above dataset.
SOURCE CODE:
>mydata$rank<-factor(mydata$rank)
>mylogit<-glm(admit~gre+gpa+rank,data=mydata,family=”binomial”)
>summary(mylogit)
OUTPUT:

```
> mydata$rank <- factor(mydata$rank)
> mylogit <- glm(admit ~ gre + gpa + rank, data = mydata, family = "binomial")
> summary(mylogit)

Call:
glm(formula = admit ~ gre + gpa + rank, family = "binomial",
    data = mydata)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6268  -0.8662  -0.6388   1.1490   2.0790

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.989979   1.139951  -3.500 0.000465 ***
gre          0.002264   0.001094   2.070 0.038465 *
gpa          0.804038   0.331819   2.423 0.015388 *
rank2       -0.675443   0.316490  -2.134 0.032829 *
rank3       -1.340204   0.345306  -3.881 0.000104 ***
rank4       -1.551464   0.417832  -3.713 0.000205 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 458.52  on 394  degrees of freedom
AIC: 470.52

Number of Fisher Scoring iterations: 4
```
1

Experiment No 8 - <u>REGRESSION MODEL FOR PREDICTION</u>

Apply regression Model techniques to predict the data on above dataset.

```
># make sure R knows region is categorical
>str(states.data$region)
 Factor w/ 4 levels "West","N. East",..: 3 1 1 3 1 1 2 3 NA 3 ...
>states.data$region<- factor(states.data$region)
> #Add region to the model
>sat.region<- lm(csat ~ region,
+                data=states.data)
> #Show the results
>coef(summary(sat.region)) # show regression coefficients table
```

Out put:

```
            Estimate Std. Error t value
Pr(>|t|) (Intercept)        946.3 14.8 63.958
1.35e-46
regionN. East    -56.8       23.1 -2.453 1.80e-02
regionSouth      -16.3       19.9 -0.819 4.17e-01
regionMidwest     63.8       21.4 2.986 4.51e-03
>anova(sat.region) # show ANOVA
table Analysis of Variance Table

Response: csat
Df Sum Sq Mean Sq F value   Pr(>F)
region     3 82049 27350      9.61
0.000049
Residuals 46 130912    2846
>
```

Experiment No.9: CLASSIFICATION MODEL

PROBLEM DEFINATION:
g. Install relevant package for classification.

SOURCE CODE:
install.packages("rpart.plot")
install.packages("tree")
install.packages("ISLR")
install.packages("rattle")

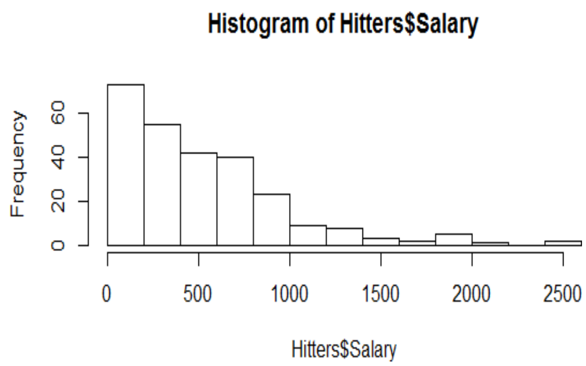
library(tree)
library(ISLR)
library(rpart.plot)
library(rattle)

PROBLEM DEFINATION:
h. Choose classifier for classification
problem. Evaluate the performance of
classifier. **SOURCE CODE:**
attach(Hit
ters)
View(Hitt
ers)
# Remove NA data
Hitters<-na.omit(Hit
ters)

# log transform Salary to make it a bit more normally distributed
hist(Hitters$Salary)

Hitters$Salary <-
log(Hitters$Salary)
hist(Hitters$Salary)

output:



SOURCE CODE:

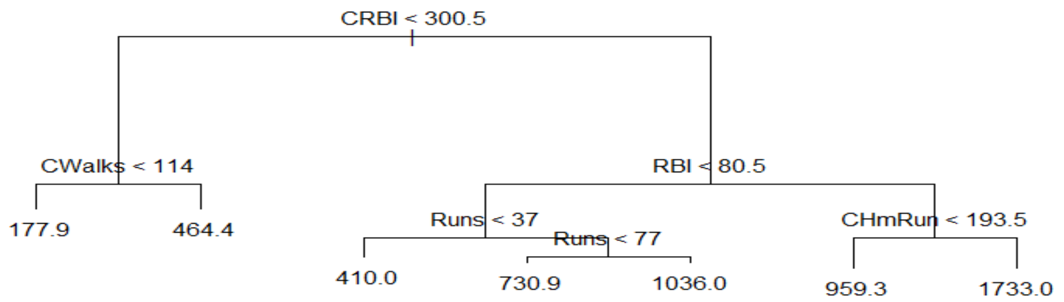
```
> tree.fit <- tree(Salary~Hits+Years, data=Hitters)
> summary(tree.fit)
```

Regression tree:
tree(formula = Salary ~ Hits + Years, data =
Hitters) Number of terminal nodes: 8
Residual mean deviance: 101200 = 25820000 /
255 Distribution of residuals:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-1238.00	-157.50	-38.84	0.00	76.83	1511.00

```
plot(tree.fit, uniform=TRUE,margin=0.2)
text(tree.fit, use.n=TRUE, all=TRUE,
cex=.8) #plot(tree.fit)
>split <- createDataPartition(y=Hitters$Salary, p=0.5, list=FALSE)
> train <- Hitters[split,]
>        test <-
Hitters[-split,]
#Create tree model
> trees <- tree(Salary~., train)
> plot(trees)
> text(trees, pretty=0)

# Cross validate to see whether pruning the tree will improve
 Performance
```

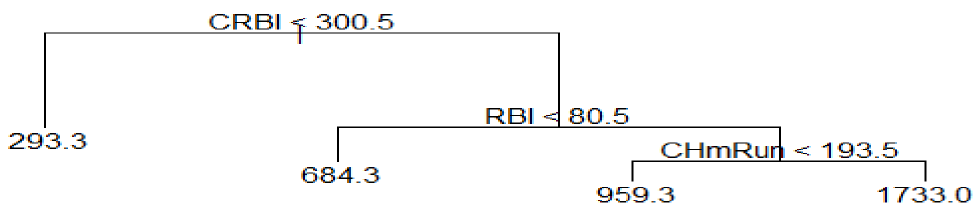
OUTPUT:



SOURCE CODE:

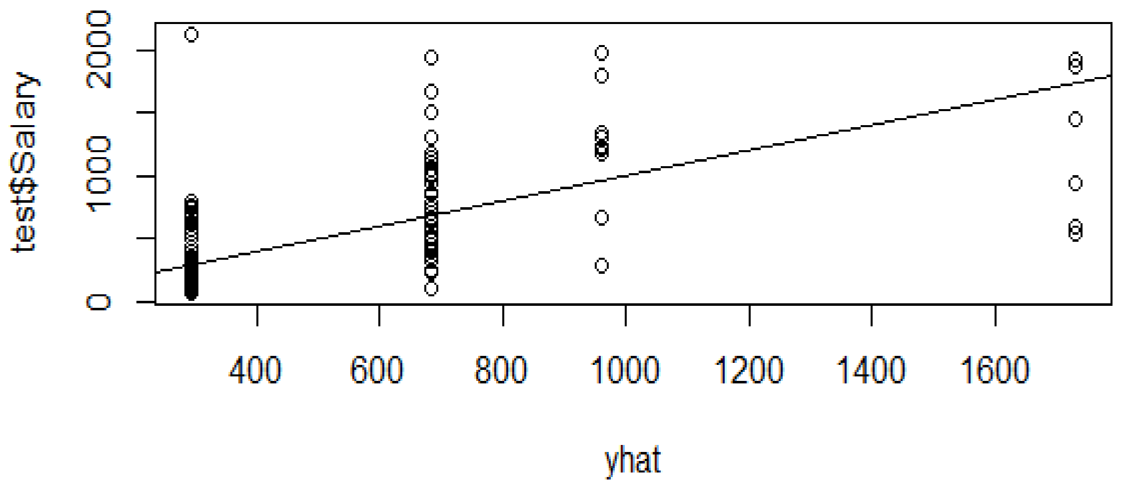
```
#Cross validate to see whether pruning the tree will improve performance
> cv.trees <- cv.tree(trees)
> plot(cv.trees)
> prune.trees <- prune.tree(trees, best=4)
> plot(prune.trees)
> text(prune.trees, pretty=0)
```

OUTPUT:



SOURCE CODE:
> yhat <- predict(prune.trees, test)
> plot(yhat, test$Salary)
> a
bline(0,1
[1]
150179.
7
> mean((yhat -
test$Salary)^2) [1]
150179.7

OUTPUT:



> mean((yhat -
test$Salary)^2) [1]
150179.7

Experiment No.10

PROBLEM DEFINATION:
CLUSTERING MODEL
e. Clustering algorithms for unsupervised
classification. Plot the cluster data using R
visualizations
SOURCE CODE:
1. Clustering algorithms for unsupervised

classification. library(cluster)

> set.seed(20)
> irisCluster <- kmeans(iris[, 3:4], 3, nstart = 20)
nstart = 20. This means that R will try 20 different random starting assignments and then select the one with the lowest within cluster variation.
> irisCluster

OUTPUT:

Petal.Length
Petal.Width 1
1.462000
0.246000
2 4.269231 1.342308
3 5.595833 2.037500

Clustering vector:

[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[42] 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2
[83] 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3
[124] 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 3 3

Within cluster sum of squares by cluster:

[1] 2.02200 13.05769 16.29167
(between_SS / total_SS = 94.3 %)

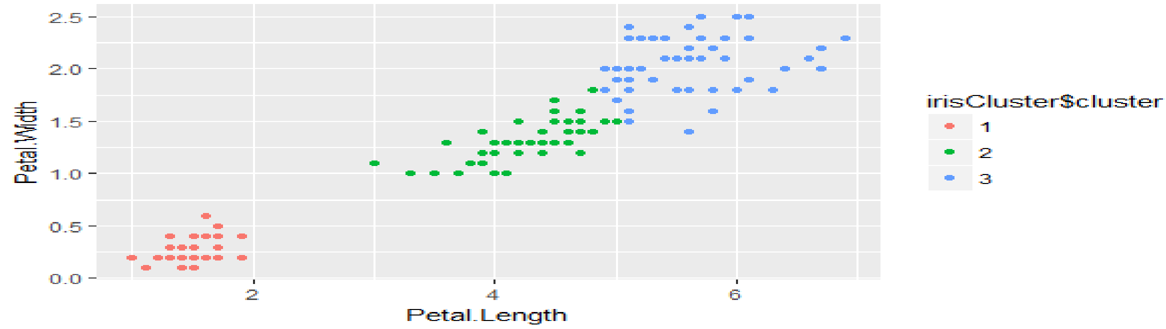
Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss"

[6] "betweenss" "size" "iter" "ifault"

SOURCE CODE:
> irisCluster$cluster <- as.factor(irisCluster$cluster)
> ggplot(iris, aes(Petal.Length, Petal.Width, color = irisCluster$cluster)) + geom_point()

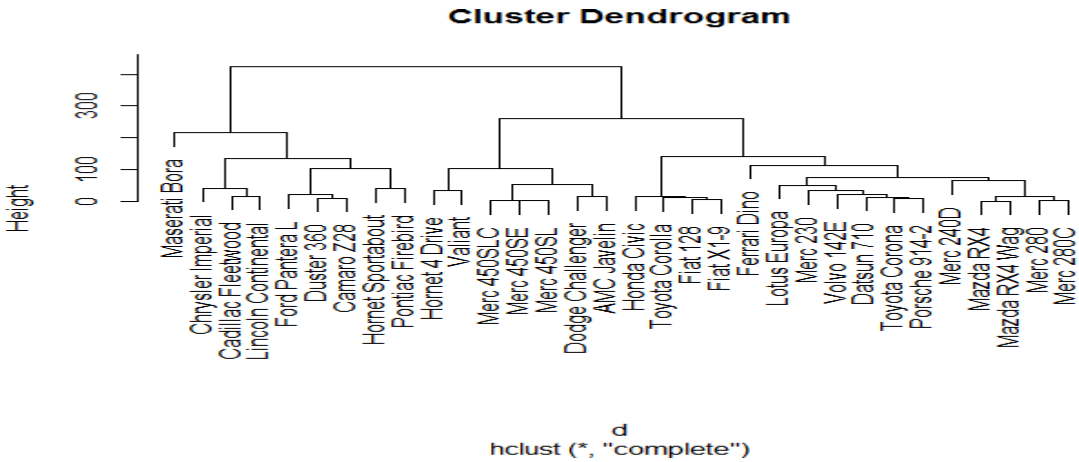
OUTPUT:



SOURCE CODE:

```
> d <- dist(as.matrix(mtcars)) # find distance matrix
> hc <- hclust(d)             # apply hirarchical clustering
> plot(hc)                    # plot the dendrogram
```

OUTPUT:

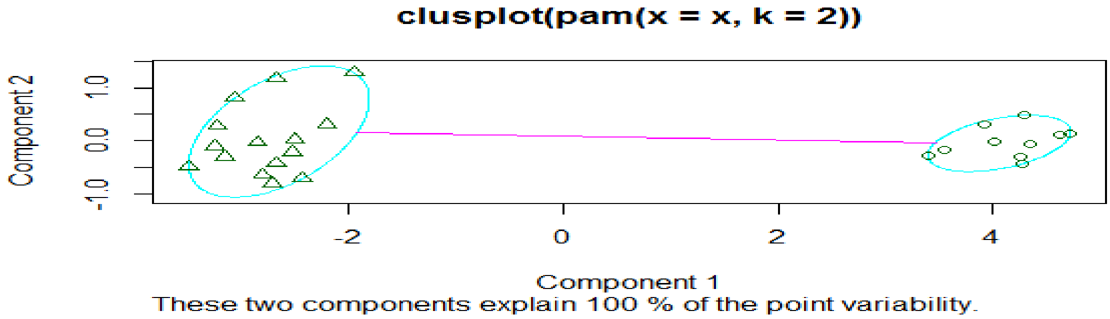


2. Plot the cluster data using R visualizations.

SOURCE CODE:

```
## generate 25 objects, divided into 2 clusters.
x <- rbind(cbind(rnorm(10,0,0.5), rnorm(10,0,0.5)),
 cbind(rnorm(15,5,0.5),
rnorm(15,5,0.5))) clusplot(pam(x, 2))
```

OUTPUT:



SOURCE CODE:

add noise, and try again :
x4 <- cbind(x, rnorm(25),
rnorm(25)) clusplot(pam(x4, 2))

OUTPUT: